Reinforcement Learning for Visual Object Detection

Stefan Mathe[2,3] **Aleksis Pirinen**[1] **Cristian Sminchisescu**[1,2]
[1]Department of Mathematics, Faculty of Engineering, Lund University
[2]Institute of Mathematics of the Romanian Academy
[3]Department of Computer Science, University of Toronto
stefan.mathe@imar.ro, aleksis.pirinen@math.lth.se, cristian.sminchisescu@math.lth.se

Abstract

*One of the most widely used strategies for visual object detection is based on exhaustive spatial hypothesis search. While methods like sliding windows have been successful and effective for many years, they are still brute-force, independent of the image content and the visual category being searched. In this paper we present principled sequential models that accumulate evidence collected at a small set of image locations in order to detect visual objects effectively. By formulating sequential search as reinforcement learning of the search policy (*including *the stopping condition), our fully trainable model can explicitly balance for each class, specifically, the conflicting goals of* exploration *– sampling more image regions for better* accuracy *–, and* exploitation *– stopping the search* efficiently *when sufficiently confident about the target's location. The methodology is general and applicable to any detector response function. We report encouraging results in the PASCAL VOC 2012 object detection test set showing that the proposed methodology achieves almost* two orders of magnitude speed-up *over sliding window methods.*

1. Introduction

Classically, detection has been formulated as the problem of maximizing a confidence function over a set of hypothesized target locations, where the confidence can be learned in a fully supervised[12] or weakly supervised[34] setup. In the sliding window formulation, the hypothesis set consists of a large set of rectangular windows, and the maximization problem is solved by exhaustive search. Since this process is generally too expensive in practice, many methods have been proposed to accelerate it, from methodologies that leverage properties of the confidence function, to proposal methods or cascade techniques. All these methods retain the exhaustive search property over the hypothesis space, aiming either to reduce the number of hypotheses to start with, or search these efficiently.

In contrast, biological systems have a pattern of search that can be characterized as 'saccade and fixate'[17], where a small set of scene locations are investigated sequentially, in order to accumulate sufficient evidence on the target location. Set aside efficiency (only a few regions of an image are explored) and biological plausibility, it appears still interesting to formally derive mathematical models that could optimally balance efficiency and accuracy, by integrating evidence, sequentially, in a principled way. The challenge is to be able to operate with delayed rewards, which rule out supervision at each step. At the same time, avoid the need to completely pre-specify the environment, which for visual scenes would be impossible – given the complexity of images and visual object categories, the models should be effectively trained.

By formulating sequential search as reinforcement learning of the category and the image-dependent search policy *including* the stopping conditions, in this work we develop fully trainable methods that can explicitly balance the conflicting goals of *exploration* – sampling more image regions for better *accuracy* –, and *exploitation* – stopping the search *efficiently* when sufficiently confident in the target's location. The methodology is general, applicable to any detector response function, and can learn search strategies and stopping conditions that are image and visual category specific. Two orders of magnitude speed-ups over sliding window methods are achieved in the challenging PASCAL VOC 2012 object detection benchmark.

2. Related Work

One class of efficient detectors focuses on the use of branch-and-bound heuristics[23] to prioritize exploration of the search space towards promising image regions. Unlike the present work, such techniques are only applicable to confidence function classes for which strong bounds are available. Additionally, in the absence of the target in the image, methods in this class degenerate to exhaustive search. Motivated by these limitations, other authors have proposed to use cascades of classifiers[39, 11, 15, 30], to progressively narrow the search space, where weak but fast classifiers are applied early to eliminate image regions unlikely to contain the target, while investing computational resources to run more complex classifiers on promising regions. Such methods drastically reduce the computation cost, but classifiers early in the cascade still have to be applied exhaustively over all image regions. Instead of focusing on region exploration strategies, other methods have sought to optimize the evaluation of the confidence function. This includes sharing computation among neighboring image regions[40, 35] or among the different classifiers for multi-class detection problems[20].

Recent trends in object detection focus on a rapid content-based reduction of the set of candidates – in earlier methods, windows, cropped at different positions, and of different aspect ratios, in an image – to a smaller set (still thousands of hypotheses in most methods) which exhibits the statistical regularities of the objects found in the real world. Typical methodologies include parametric figure-ground segmentation with Gestalt, ‘object-like’ filtering[5], superpixels[38, 32] or edge-based cues[21]. In this work we will rely on the parametric segmentation method of Carreira *et al.*[5] to generate a set of free-form figure-ground proposals that capture most objects of interest, although our method can use any other state of the art proposal generation method[38, 32, 21].

In contrast to methods based on branch and bound and cascades of classifiers, sequential search methods like [13] attempt to sparsely sample the image through a local search guided by the contextual relations among regions, previously shown to improve detection accuracy[6, 7, 16, 31]. [13] propose search policies that map contextual windows to the ground truth target location based on random forests, whereas [37, 9] learn a mapping from images to bounding box masks using a cascaded deep learning model.

Palleta *et al.*[25] and Butko and Movellan[3] developed remarkable early sequential models based on POMDPs for recognition and face detection. However, those models are not fully trainable and require a complete and accurate specification of the environment, which makes them challenging to apply in complex multi-class visual detection setups. More recently, reinforcement learning[36] has been applied to visual analysis problems like image classification[24, 19, 29], face detection[14], tracking and recognizing objects in video[2], learning a sequential policy for RGB-D semantic segmentation[1], or scanpath prediction[27].

In independent work performed in parallel with ours[28], [4] also focus on object detection using reinforcement Q-learning. We differ, among others, in using policy search based on an analytic gradient computation with continuous as opposed to discrete reward (both in a supervised and weakly-supervised image labeling setup[26]), by operating on regions instead of deforming bounding boxes, in using different actions (infinite set via function approximation vs. 9 in [4]), a different state representation (a set of 10 boxes in [4] vs. our manipulation of disjoint sets), and in the training procedure based on reinforcement learning with delayed rewards as opposed to an additional apprenticeship signal in [4]. This results in a different model behavior in both training and testing, as [4] requires the control of actions via short steps in order to prevent the apprenticeship learning process to immediately locate the target from any position. In testing [4] use 10 steps to locate the target, whereas our model takes 3.1 steps on average.

Relevant to our work is also the one of Karayev et al.[18] who differently however, focus on object detection in an anytime recognition framework where a multi-class detector can be stopped, asynchronously, during its execution. [18] sequentially schedule multi-class models, optimizing the order of applying sliding window object detectors (exhaustively evaluated at all image locations, in a cascade), stopping short of running detectors for some classes. In contrast, we spatially optimize each specific sequential class detector (stopping short of searching all image locations) and run the detectors for all classes in the standard way. Methodologically, there are significant differences: [18] use Q-learning and regress expected value of (state, action), we do policy search with analytic gradient to directly optimize expected reward. We have infinite action spaces (any image location), [18] operate over finite actions (1+#detector-classes in [18], or 1+#feature-types in [19, 8]); [18] can stop anytime, whereas we learn a stopping condition for each class. From a system viewpoint, the methods are complementary, as one can benefit both from an efficient ordering of class detectors[18] and from efficient individual class detectors, as we propose, but we will not investigate this here.

3. Problem Formulation

Given an input image, we formulate action detection as the problem of maximizing a confidence function $f_c : R \rightarrow \mathbb{R}$ over the set of image regions R:

$$\mathbf{r}^* = \arg\max_{\mathbf{r} \in R} f_c(\mathbf{r}) \tag{1}$$

The set of image regions R can be defined either at the coarse level of bounding boxes or at the finer level of free-form image regions obtained with a state of the art proposal generation method[5, 38, 32].

Good choices for the confidence function f_c that achieve state-of-the-art performance, are associated with a high computational price tag. Therefore, solving the optimization problem (1) can still be expensive even for the comparatively smaller (versus e.g. bounding boxes) set of region proposals R obtained by a segmentation algorithm. To address this issue, in §3.1 we present a model to learn efficient search strategies, rigorously formulated in a reinforcement

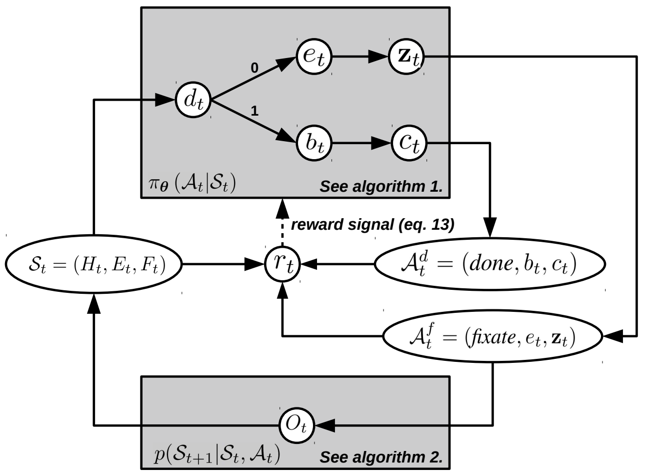
0
1
e_t
$\mathbf{z}_t$
d_t
b_t
c_t
$\pi_\theta(\mathcal{A}_t|\mathcal{S}_t)$
See algorithm 1.
reward signal (eq. 13)
$\mathcal{S}_t = (H_t, E_t, F_t)$
r_t
$\mathcal{A}_t^d = (done, b_t, c_t)$
$\mathcal{A}_t^f = (fixate, e_t, \mathbf{z}_t)$
O_t
$p(\mathcal{S}_{t+1}|\mathcal{S}_t, \mathcal{A}_t)$
See algorithm 2.

Figure 1. Sequential detector based on reinforcement learning. At each time step, the model may terminate search ($d_t = 1$) based on the history H_t of observed regions (Algorithm 1) and produce a detection hypothesis b_t with confidence c_t, receiving a reward measuring the detection quality. Otherwise, an evidence region e_t is chosen from the set $H_t \setminus E_t$ of unselected regions and used to predict the next fixation location $\mathbf{z}_t$. The set O_t of all regions in the neighbourhood of $\mathbf{z}_t$ become observed (Algorithm 2) and a negative reward is received, reflecting the computational cost of extracting features for these regions.

learning setup. Our model operates in an integrate, fixate and evaluate regime, and only explores a few locations before deciding on the presence of a target.

3.1. Sequential Detection Model

In this section, we present the key components of our optimal sequential model for image exploration.[1] Our model is given a set of image regions R indexed by the set $B = \{1, \ldots, |R|\}$ (with $|\cdot|$ the set cardinality), the confidence function as introduced in (1), $f_c(\mathbf{r}) = \boldsymbol{\theta}_c^\top \mathbf{q}(\mathbf{r})$ with parameters $\boldsymbol{\theta}_c$, and a feature extractor $\mathbf{q} : R \to \mathbb{R}^m$ of dimensionality m. The objective of the model is to locate the target with a minimal number of evaluations of these two computationally expensive functions

At each time step t during a detection sequence (except the last step), our model generates a *fixate* action $\mathcal{A}_t^f$, based on its internal state $\mathcal{S}_t$. Each fixation action specifies a location in the image that the model decides to explore, and results in a set of observations O_t, which is a set of image regions in the proximity of this location. The observed regions are the only regions that are inspected by the algorithm. In particular, they are the only regions on which the confidence function f_c and feature extractor $\mathbf{q}$ need to be evaluated. The observations O_t are then used to update the state $\mathcal{S}_t$, summarizing all past observations and actions.

When enough information has been collected about the image, the model issues a special *done* action, indicating that it has decided on the location of the detection target. The *done* action is associated with a detection target bounding box b_t and a detection confidence value c_t. The model has a set of trainable parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_c, \boldsymbol{\theta}_d, \boldsymbol{\theta}_e, \boldsymbol{\theta}_p, \Sigma_p, \sigma_c)$ controlling, respectively, the detector response confidence, the stopping criteria, the informativeness of an image region with respect to the target location, the image location of the most probable next fixation and its variance, and the variance of the confidence c_t associated to the model output.

Each *fixate* action $\mathcal{A}_t^f$ can potentially reduce the uncertainty in localizing the detection target, but is associated with a computational cost due to the need to integrate the set of observations O_t into the state. The goal of our model is to balance the conflicting needs of information gathering (*fixate* actions) with the need to correctly locate the target (*done* actions).

3.2. Model Structure

We now proceed to describe in detail the actions, states, observations and the decision process of our model. The model components are shown in fig. 1, and several examples of search patterns are illustrated in fig. 2.

States: The state of our model is represented as a tuple with three elements: the observed region history H_t, the selected evidence region history E_t and the fixation history F_t. This tuple $\mathcal{S}_t = (H_t, E_t, F_t)$ summarizes the history of observations and actions since the beginning of the search sequence.

The observed region history H_t: At each time step t, the model keeps track of a history $H_t \subseteq B$ of image regions observed so far. The confidence function f_c is evaluated on these regions alone and used to decide when to terminate the search. The history H_t is also used by the model to decide on promising locations to fixate during the next step, as these might provide context to guide the search.

The selected evidence region history E_t: The model decides on the next location to fixate based on an evidence region $e_t \in H_t$ from the observation history. This evidence region is deemed by the model to provide the necessary context that is indicative of the target's location. However, to encourage diversity during search, each region should be used as evidence at most once. For this reason, the model keeps track of the set $E_t \subseteq H_t$ of regions selected so far, and evidence regions are always selected from the set $H_t \setminus E_t$.

The fixation history F_t: The set of observed regions at

[1]Please see our accompanying report[26] for detailed derivations.

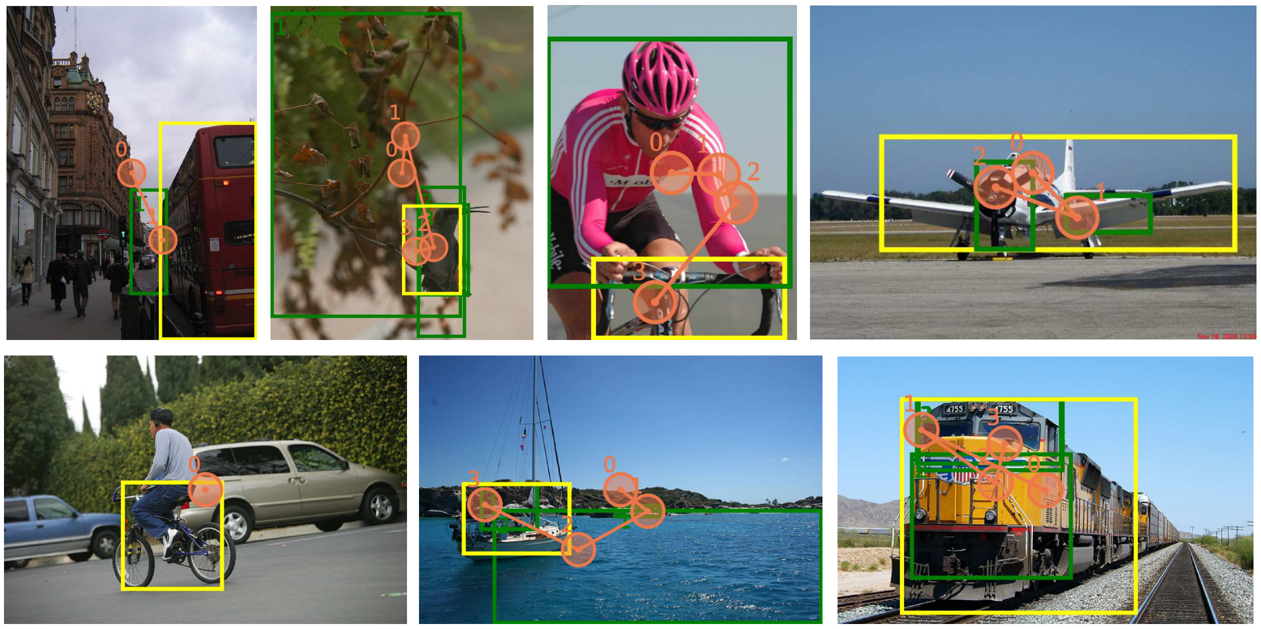
Figure 2. Sequences of fixation locations $\mathbf{z}_t$ (orange circles) generated by our model, together with the corresponding evidence regions e_t (green boxes), and the final detected bounding box b_t (yellow), for several images. The model may terminate the search early, if the target is found by the first central fixation (first image in the second row). When the target has not yet been found, regions that do not contain it are often exploited to guide the search to new promising locations (*e.g.* the street provides the context for finding the bus). When a small target lies inside a wider region (*e.g.* the bird in the tree), the model uses the wider region as a contextual cue to find the target, in a coarse to fine fashion. Similarly, fine-to-coarse search strategies involving several exploratory fixations are used to provide the foveal coverage needed to observe large targets (*e.g.* the airplane). See Table 1 for quantitative results and §4 for discussion.

each time step t depends on the history F_t of past fixation locations, *c.f.* (2). We therefore include this history into the state $\mathcal{S}_t$.

Actions: Actions in our model are represented as tuples. There are two kinds of actions, distinguished by their first element, which can be one of two discrete symbols: *fixate* or *done*.

Fixate actions are represented as a three element tuple $\mathcal{A}_t^f = (\textit{fixate}, e_t, \mathbf{z}_t)$, where $e_t \in B$ represents the index of the evidence region and $\mathbf{z}_t \in \mathbb{R}^2$ is the image coordinate of the next fixation. *Done* actions are represented as $\mathcal{A}_t^d = (\textit{done}, b_t, c_t)$ where $b_t \in B$ is the index of the region representing the detection output and $c_t \in \mathbb{R}$ represents the detection confidence. To summarize, the action space of our model consists of the union of all *fixate* and *done* tuples, *i.e.* $\mathcal{A} = \mathcal{A}^f \cup \mathcal{A}^d$.

Observations: Following a *fixate* action, the set O_t of image regions in the neighbourhood of the fixation location $\mathbf{z}_t$ become observed.

To define this neighbourhood, we use a circular area of radius T_R around the fixation center $\mathbf{z}_t$. We say that a pixel is fixated at time t if it falls within the area associated with $\mathbf{z}_t$. In order for a region $\mathbf{r}$ to become observed at time t, a sufficiently large fraction $h(\mathbf{r})$ of its pixels must have been fixated during the current or previous steps:

$$h(\mathbf{r}) = \frac{|\{\mathbf{x} \in \mathbf{r} \,|\, (\exists)\, \mathbf{z} \in F_t, \|\mathbf{z} - \mathbf{x}\|_2 \leq T_R\}|}{|\mathbf{r}|} \quad (2)$$

$$O_t = \{i \in B \mid h(\mathbf{r}_i) \geq T_F\} \quad (3)$$

where $F_t = \{\mathbf{z}_1, .., \mathbf{z}_t\}$ is the history of locations fixated by the model up to time step t, and T_F is a threshold that controls the minimum fraction of fixated pixels in an observed segment.

3.3. Stochastic Policy

The model decides on the next action to take based on the current state. Its stochastic decision policy $\pi_{\boldsymbol{\theta}}(\mathcal{S}_t, \mathcal{A}_t)$ proceeds in three phases, each having its own set of learned parameters. The model evaluates whether to *terminate search (termination decision)*. If positive, a *done* action is performed, else a *fixate* action follows. We will review each of these, next.

Termination decision: The model may decide to terminate search at any given time step, based on the current state $\mathcal{S}_t$, and produce a detection result. Rather than using an ad-hoc termination policy, *e.g.* a preset number of fixations (search locations), our model uses a learnt decision function that balances detection confidence against computational load:

- *Detection confidence:* If the model has already observed a region which is deemed to contain the detection target with high confidence, it may decide to terminate the search early. To capture this aspect, we compute the maximum confidence over the regions observed so far, *i.e.* $\operatorname{asmax}\left(\{f_c(\mathbf{r}_i)\}_{i\in H_t}\right)$, where $\operatorname{asmax}(X) = \frac{\sum_{x\in X} x e^{\alpha x}}{\sum_{x\in X} e^{\alpha x}}$ for any set X and smoothness meta-parameter α.

- *Computational load:* The running cost of our detector has two components: first, the number of confidence function evaluations performed so far, which is proportional to the ratio $|H_t| / |R|$ of regions observed at the current time step t; second, the number of search policy evaluations. Since the policy is evaluated once per time step, this cost is proportional to the number of time steps t.

In order to allow the model to balance these termination criteria, we define a four-element feature vector for the current state:

$$\mathbf{v}(\mathcal{S}_t) = \begin{bmatrix}\operatorname{asmax}\left(\{f_c(\mathbf{r}_i)\}_{i\in H_t}\right) & t & \frac{|H_t|}{|R|} & 1\end{bmatrix}^\top \tag{4}$$

The search termination probability (*done* action) is given by a logistic classifier with parameters $\boldsymbol{\theta}_\mathrm{d}$:

$$p_{\boldsymbol{\theta}}(d_t = 1|\mathcal{S}_t) = \operatorname{sigm}\left[\boldsymbol{\theta}_\mathrm{d}^\top \mathbf{v}(\mathcal{S}_\mathbf{t})\right] \tag{5}$$

where d_t is an binary variable indicating the decision to terminate the search at the current time step and $\operatorname{sigm}(x) = (1+e^{-x})^{-1}$ is the sigmoid function.

Done action: Upon termination ($d_t = 1$), the model outputs a bounding box b_t from the set H_t of observed regions, to represent the detection target location, together with a confidence score c_t. We use a soft maximum bounding box selection criterion, with smoothness meta-parameter α:

$$p_{\boldsymbol{\theta}}(b_t = k|d_t = 1, \mathcal{S}_t) = \frac{e^{\alpha f_c(\mathbf{r}_k)}}{\sum_{i\in H_t} e^{\alpha f_c(\mathbf{r}_i)}} \tag{6}$$

The corresponding confidence c_t is normally distributed around the confidence for the selected bounding box:

$$p_{\boldsymbol{\theta}}(c_t|d_t = 1, b_t = k, \mathcal{S}_t) = N(c_t|f_c(\mathbf{r}_k), \sigma_c) \tag{7}$$

where $\sigma_c \in \mathbb{R}$ is a model parameter that controls the variance of the confidence predictions.

Finally, the probability of a *done* action is given by:

$$\begin{aligned}\pi_{\boldsymbol{\theta}}\left(\mathcal{A}_t = (\text{done}, b_t, c_t)\,|\mathcal{S}_t\right) = p_{\boldsymbol{\theta}}\left(d_t = 1|\mathcal{S}_t\right)\cdot \\ \cdot\, p_{\boldsymbol{\theta}}(b_t|d_t = 1, \mathcal{S}_t) p_{\boldsymbol{\theta}}(c_t|d_t = 1, b_t, \mathcal{S}_t)\end{aligned} \tag{8}$$

Fixate action: If the search is not terminated ($d_t = 0$), the model selects a new evidence region $e_t \in (H_t \setminus E_t)$ from the set of observed regions, that it deems informative for the target location.

We define an evidence function $f_\mathrm{e} : B \to \mathbb{R}$, $f_\mathrm{e}(i) = \exp\left[\boldsymbol{\theta}_\mathrm{e}^\top \mathbf{q}(\mathbf{r}_i)\right]$ that evaluates the informativeness of image region i with respect to the target location, where $\boldsymbol{\theta}_\mathrm{e}$ are learned model parameters. We pick the region e_t from a multinomial distribution defined by the evidence function over the set $H_t \setminus E_t$ of image regions not selected during previous steps:

$$p_{\boldsymbol{\theta}}(e_t|\mathcal{S}_t) = \frac{f_\mathrm{e}(e_t)}{\sum_{i\in H_t\setminus E_t} f_\mathrm{e}(i)} \tag{9}$$

Once selected, the evidence region e_t is used to define a Gaussian probability distribution for the next fixation location $\mathbf{z}_t \in \mathbb{R}^2$. For convenience, let us denote by

$$\boldsymbol{\mu}(e_t) = \frac{\mathbf{x}_1(e_t) + \mathbf{x}_2(e_t)}{2} \tag{10}$$

the center of the bounding box tightly enclosing the evidence region $\mathbf{r}_{e_t}$, defined by its top-left and bottom-right corners $\mathbf{x}_1(e_t)$ and $\mathbf{x}_2(e_t)$, respectively. Similarly, let

$$\boldsymbol{\Delta}(e_t) = \operatorname{diag}\left(\frac{\mathbf{x}_1(e_t) - \mathbf{x}_2(e_t)}{2}\right) \tag{11}$$

be the diagonal matrix encoding half the width and height of this bounding box. Then, the probability for the next fixation location $\mathbf{z}_t$ is:

$$p_{\boldsymbol{\theta}}(\mathbf{z}_t|\mathcal{S}_t, e_t) = N\left(\cdot|\mathbf{f}_\mathrm{p}(e_t), \boldsymbol{\Delta}(e_t)^\top \Sigma_p \boldsymbol{\Delta}(e_t)\right) \tag{12}$$

where Σ_p is a learned covariance matrix that controls the spread of fixations, and the Gaussian center $\mathbf{f}_\mathrm{p}(e_t)$ is based on a linear combination of the evidence region features $\mathbf{q}(\mathbf{r}_{e_t})$ with learned parameters $\boldsymbol{\theta}_\mathrm{p}$:

$$\mathbf{f}_\mathrm{p}(e_t) = \boldsymbol{\Delta}(e_t)\boldsymbol{\theta}_\mathrm{p}^\top \mathbf{q}(e_t) + \boldsymbol{\mu}(e_t) \tag{13}$$

We make the position function invariant to the scale of the image region $\mathbf{r}_i$ by normalizing with respect to its bounding box size, defined by the top-left and bottom-right corners *c.f.* first term in (13), and relative to the bounding box center (second term in (13)).

Summarizing, the probability of a *fixate* action is given by:

$$\begin{aligned}\pi_{\boldsymbol{\theta}}\left(\mathcal{A}_t = (\text{fixate}, e_t, \mathbf{z}_t)\,|\mathcal{S}_t\right) = \\ = p_{\boldsymbol{\theta}}\left(d_t = 0|\mathcal{S}_t\right) p_{\boldsymbol{\theta}}\left(e_t|\mathcal{S}_t\right) p_{\boldsymbol{\theta}}\left(\mathbf{z}_t|\mathcal{S}_t, e_t\right)\end{aligned} \tag{14}$$

Algorithm 1 Policy sampling algorithm

```
1:  procedure SAMPLE ($\mathcal{S}_t = (H_t, E_t, F_t)$)
2:      $d_t \sim p(d_t|\mathcal{S}_t)$ using (4), (5)
3:      if $d_t = 1$ then
4:          $b_t \sim p(b_t|\mathcal{S}_t, d_t)$ using (6).
5:          $c_t \sim p(c_t|\mathcal{S}_t, d_t, b_t)$ using (7)
6:          return $\mathcal{A}_t = (\text{done}, b_t, c_t)$
7:      else
8:          $e_t \sim p(e_t|\mathcal{S}_t, d_t)$ using (9)
9:          $\mathbf{z}_t \sim p(\mathbf{z}_t|\mathcal{S}_t, d_t, e_t)$ using (12)
10:         return $\mathcal{A}_t = (\text{fixate}, e_t, \mathbf{z}_t)$
11:     end if
12: end procedure
```

Algorithm 2 State transition algorithm

```
1:  procedure OBSERVE ($\mathcal{S}_t = (H_t, E_t, F_t)$, $\mathcal{A}_t = (\text{fixate}, e_t, \mathbf{z}_t)$)
2:      $O_t \leftarrow \{i \in B \mid h(\mathbf{r}_i) \geq T_F\}$
3:      $H_{t+1} \leftarrow H_t \cup O_t$
4:      $E_{t+1} \leftarrow E_t \cup \{e_t\}$
5:      $F_{t+1} \leftarrow F_t \cup \{\mathbf{z}_t\}$
6:      return $\mathcal{S}_{t+1} = (H_{t+1}, E_{t+1}, F_{t+1})$
7:  end procedure
```

The model policy is completely specified by equations (8), (14), which define a probability distribution over all possible actions $\mathcal{A}_t$. Notice that out policy is highly (deeply) non-linear in the features and the parameters. The stochastic policy is given by a Gaussian distribution on top of highly non-linear predictions (in contrast notice that methodologies like [14, 4] are deterministic).

3.4. Inference and Learning

Inference is carried out by repeated sampling of the policy $\pi_{\boldsymbol{\theta}}(\mathcal{A}_t|\mathcal{S}_t)$, until a *done* action is achieved (Algorithm 1). At each step the state $\mathcal{S}_t$ is updated according to the action $\mathcal{A}_t$ (Algorithm 2). When the search is finished, the region b_t and the confidence c_t are generated and returned as the detector output.

For **learning** we are given a set of images, represented as sets of regions B_j, together with confidence function f_c aimed to be maximal at target locations. For notational simplicity, without loss of generality, we will consider the equations for one image, containing n (possibly 0) detection targets, and the corresponding ground truth regions $\{\mathbf{g}_i\}_{i=1}^n$.

We wish to find the model parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_\text{c}, \boldsymbol{\theta}_\text{d}, \boldsymbol{\theta}_\text{e}, \boldsymbol{\theta}_\text{p}, \Sigma_\text{p}, \sigma_\text{c})$ maximizing the target detection accuracy based on the detected target location b_t and confidence c_t at the last step (when $d_t = 1$). At the same time, we aim to minimize the number of region evaluations. To capture the trade-off, and to avoid explicitly instructing the model how to achieve it, we formulate the training objective as a delayed reward, as typical in a reinforcement learning setup. Our reward function is sensitive to the detection location and the confidence at the final state, and incurs a penalty for each region evaluation:

$$r_t\left(\mathcal{S}_t, \mathcal{A}_t; \{\mathbf{g}_i\}_{i=1}^n\right) = \\ = \begin{cases} -\beta \cdot |O_t \setminus H_t| & \text{if } d_t = 0 \\ \text{sigm}\left(c_t\right) \cdot \left[\max_{i=1,n} \text{iou}\left(\mathbf{g}_i, \mathbf{r}_{b_t}\right)\right] & \text{if } d_t = 1 \wedge n > 0 \\ -\text{sigm}\left(c_t\right) & \text{if } d_t = 1 \wedge n = 0 \end{cases} \tag{15}$$

where $\text{iou}(\cdot,\cdot)$ is the intersection over union function on regions and β is a penalty paid by the model for each confidence function evaluation.We found it straightforward to estimate the exploitation-exploration trade-off parameters, for each class detector, based on cross-validation. Typical values are *e.g.* $\beta = 10^{-3}$ and $\alpha = 30$.

The first branch associates a negative reward to each *fixate* action, proportional to the computational cost of evaluating the newly observed region set $O_t \setminus H_t$. The last two branches correspond to the *done* action, with different rewards for images in which the target is present and absent. In the former case (branch 2), the model receives a reward that is proportional to its confidence and the ground truth overlap. In the latter case (branch 3), the location is ignored, and the model receives a higher reward if its confidence is smaller. Concluding, the reward function defined in (15) balances detection accuracy and computational complexity.

During training, we maximize the expected reward function on the training set, defined as:

$$F(\boldsymbol{\theta}) = \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{s})}\left[\sum_{t=1}^{|\mathbf{s}|} r_t\right] - \frac{\lambda}{2}\boldsymbol{\theta}^\top\boldsymbol{\theta} \tag{16}$$

where $\mathbf{s} = ((\mathcal{S}_0, \mathcal{A}_0), \ldots, (\mathcal{S}_k, \mathcal{A}_k), \ldots)$ represents a variable length sequence of states[2], sampled by running the model (Algorithm 1 and 2), starting from an initial state $\mathcal{S}_0 = (H_0, E_0, F_0)$ and λ is an L2 regularizer. We set the initial H_0 to the set of segments observed by fixating the image center, and both E_0 and F_0 to $\emptyset$.

For one image, the gradient of the expected reward can be approximated as[41, 36]:

$$\nabla_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}\sum_{t=1}^{|\mathbf{s}^i|}\nabla_{\boldsymbol{\theta}}\text{log}\pi_{\boldsymbol{\theta}}(\mathcal{A}_t^i|\mathcal{S}_t^i)\left[\sum_{t=1}^{|\mathbf{s}^i|} r_t^i\right] + \lambda\boldsymbol{\theta} \tag{17}$$

where $\mathbf{s}^i = ((\mathcal{S}_0^i, \mathcal{A}_0^i), \ldots, (\mathcal{S}_k^i, \mathcal{A}_k^i), \ldots)$ and r_t^i, represent sequences of states, actions and corresponding re-

[2]As the model decides when to terminate search, individually, for each search path.

wards, sampled by model simulation (total of M sampled sequences).

Training our sequential model involves computing the expected reward and its gradient, *c.f.* (17),(16). For each image, this involves simulating the model until the search is terminated, by generating sequences in the state-action space. At each time step t, an action $\mathcal{A}_t$ is sampled from the policy, using Algorithm 1. More precisely, first the distribution $p_{\boldsymbol{\theta}}(d_t|\mathcal{S}_t)$ is sampled to decide whether the search is to be terminated (*done* action, i.e. $d_t = 1$). If so, then the output region index b_t and the confidence c_t are sampled from $p_{\boldsymbol{\theta}}(b_t|d_t = 1, \mathcal{S}_t)$ and $p_{\boldsymbol{\theta}}(c_t|d_t = 1, \mathcal{S}_t)$, respectively. Otherwise (not *done*, i.e. $d_t = 0$), an evidence region is selected by sampling $p_{\boldsymbol{\theta}}(e_t|d_t = 0, \mathcal{S}_t)$, and then the next fixation location is sampled from $p_{\boldsymbol{\theta}}(\mathbf{z}_t|d_t = 0, e_t, \mathcal{S}_t)$. Finally, the state of the model is updated, as described in Algorithm 2. Multiple sample sequences are generated in this way, for each image, and used to estimate the expectations.[3]

4. Experiments and Results

In this section, we present experiments to validate our search method on the challenging Pascal VOC 2012 object detection Benchmark[10], over the withheld test set available via the evaluation server. In most of our experiments, the region space R consists of all segments extracted using a figure-ground region proposal method, and any state of the art method applies. Without loss of generality, we select the CPMC algorithm[5] as their segments can be mapped with reasonable accuracy to detection targets (according to our studies, the average intersection over union overlap of the best segment enclosing rectangle with the ground truth bounding box, is 0.687).

Pipelines: To quantify the performance of different standard search models, we either solve the maximization problem (1) exactly, by performing exhaustive sliding window search (SW), exhaustive search over the CPMC region proposal set (RP), or by using our sequential reinforcement learning search model (RL).

Experimental procedure: We now present and discuss the details of our experiments.

Proposal generation: To obtain our RP hypotheses, we run the public implementation [5] over the input image. For the sliding window (SW) baseline, region hypotheses are windows obtained by iterating over various window sizes and aspect ratios, and, for each scale and aspect ratio setting, by sliding the window with a fixed stride over the image. Our sliding window enumeration strategy results in 25,000 windows per image. For region proposal, we use an optimized version of CPMC, which operates on a reduced search space formed by free-form regions.[4]

Feature extraction: For our feature extractor $\mathbf{q}$, we use the deep neural network of Krizhevsky *et al.*[22]. For a region, we invoke the network over the contents of the bounding box, and to capture context, on the entire image where the bounding box has been masked out (filled by its mean color). For each neural network evaluation, we record the output of the last fully connected layer. We concatenate the resulting feature vector with a representation of the bounding box size and aspect ratio, and obtain a final vector of 8204 values. Deep neural networks can be refined to further increase detection accuracy[12, 33]. In this work we have focused on optimal search models and have therefore opted to illustrate our model with a simpler linear SVM model trained using a generic feature extractor[22]. Note however that our method is sufficiently general to operate in conjunction with any confidence function.

Training the sequential reinforcement learning detector: We find the optimal parameter vector $\boldsymbol{\theta}$ that maximizes the expected reward (16) on the training set of the Pascal VOC 2012 Object detection challenge, using a BFGS optimizer. However, due to the high number of parameters, the model is prone to overfit the data. Therefore, in practice, we have chosen to initialize our confidence function parameters $\boldsymbol{\theta}_{\text{c}}$ by pre-training using a linear SVM where positive instances are ground truth bounding boxes and negative instances are sampled from other image locations (from the region proposal set R).

We initialize $\boldsymbol{\theta}_p$ by performing a regression from image regions to the centers of ground truth bounding boxes. We bias $\boldsymbol{\theta}_{\text{c}}$ towards their initial values while the rest of the parameters *i.e.* $(\boldsymbol{\theta}_{\text{d}}, \boldsymbol{\theta}_{\text{e}}, \sigma_{\text{c}}, \Sigma_{\text{p}})$ are initialized by uniform random sampling, in the range $[0, 1]$ and optimized using a 0 mean quadratic penalty *c.f.* (16). We validate the observation model parameters T_R as in (2) and T_F as in (3) on the Pascal VOC validation set, setting them to 64 pixels and 0.25, respectively. In practice the sensitivity associated to these parameters is not high: even if the model runs for several fixations, only a small fraction of the the total number of regions $|R|$ is observed. Empirically, we found the model to produce fairly short and effective search patterns with a number of 3.1 image locations inspected on average. As our policy is stochastic, multiple object instances can be found. Moreover, in evaluation, all visited regions above a threshold (e.g. all attended regions) are identified (locate and restart strategies are also possible).

Computational efficiency and accuracy: The running time and accuracy of our method is shown in table 1.

[3]For a training set of images, we will naturally aggregate (sum over) such estimates, for each image.

[4]Notice however that the optimization only applies to the segment generation step. Therefore, as of recent trends in region proposal-based detection, we work with larger pools typically having thousands of segments, and avoid the expensive segment filtering and ranking steps.

method	metric	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor	mean
SW	**detection AP (%)**	49.2	46.1	21.8	12.8	6.7	46.8	25.4	50.4	9.4	27.1	21.3	47.9	39.5	47.7	22.4	10.9	26.4	25.1	45.1	41.4	**31.2**
RP	**detection AP (%)**	44.5	36.3	28.0	14.4	3.6	44.7	27.0	57.6	8.8	26.6	20.2	47.7	39.7	45.1	22.6	8.7	25.6	23.6	42.1	39.2	**30.3**
RL	**detection AP (%)**	47.4	31.4	21.0	9.5	2.5	44.7	19.4	50.3	6.1	18.1	21.1	46.8	35.8	40.4	18.7	8.5	17.8	18.6	41.5	38.8	**27.0**
	evaluated regions	102	105	110	109	119	99	115	103	120	98	112	106	113	101	107	111	105	110	103	102	**107**
	running time (s)	37.6	38.8	40.0	39.8	41.9	36.4	40.8	37.1	42.3	36.6	39.9	38.0	40.2	36.5	38.2	39.5	37.6	39.3	36.9	37.0	**38.7**
	speedup (SW)	69.4	69.3	67.3	67.6	64.2	73.3	66.0	72.5	63.6	71.2	67.4	70.9	66.9	73.7	70.3	68.1	71.6	68.4	72.9	72.8	**69.6**
	speedup (RP)	8.3	8.1	7.9	7.8	7.3	8.5	7.5	8.3	7.3	8.6	7.7	8.1	7.6	8.4	8.0	7.8	8.1	7.8	8.2	8.3	**8.0**

Table 1. Detection accuracy (reported as Average Precision, AP) and running times of different methods in the test set associated to the PASCAL VOC 2012 object detection benchmark. Shown are results for the proposed sequential detection model (RL) as well as the classical sliding window (SW) and region proposal (RP) approach. The average running time of our sliding window baseline is $2,691$ seconds, regardless of the class. In the current experiments, we chose to optimize speed-up (two orders of magnitude), but our method can also be tuned for accuracy *c.f.* (15). For instance, similar accuracy with exhaustive search methods can be achieved with a 18x speed-up.

The accuracy of our sequential detector is close to that of the much more expensive sliding window baseline, although it is more than 70 times faster on an Intel Xeon 2.2Ghz CPU. This speedup takes into account the overhead of the RP algorithm (6.1 seconds) and the small overhead needed to sample the policy of our sequential detector (32 ms). We explicitly chose to give speed-ups in running times (as opposed to *e.g.* number of inspected locations or detector evaluations) as these also cover the overheads (*e.g.* in our case the additional work for the segment proposal generation step or estimating the next action), for fair comparisons with sliding windows or region proposal methods.

Besides comparisons with the SW and RP baselines, presented in table 1, it could be useful to relate to other efficient search methods like [13]. As code is not available and there are quite significant methodological, as well as region and feature representation differences, one can still consider overall speed-ups reported for similar datasets. For example, by operating over free-form regions obtained from selective search[38], [13] achieve a 9x acceleration, respectively, over sliding-windows methods in the PASCAL VOC Object Detection 2007 dataset. Both us and [13] could additionally benefit from embedding our accelerated spatial class detectors into the complementary, effective multi-class detector scheduling mechanism for anytime recognition proposed in [18]. This could further produce a 2x speed-up at roughly similar AP loss.

Qualitative analysis: We note that the length of the search sequence is greatly dependent on the image (see fig. 2). If the target is close to the image center (*e.g.* the bicycle in fig. 2), the method tends to terminate the search after the first fixation, as it has already confidently located the object. If the target is located near the periphery, our model tends to continue the search over a longer time horizon. This behavior illustrates the model's capacity to adapt the search sequence length to the input image, as opposed to other fixed-lenght search methods in the literature.

Our visualizations of the results reveal three ways in which an evidence region (shown in green in fig. 2) may guide the search: *(a)* The contextual region may not contain the target, but instead provide cues on its location (*e.g.* the ocean for the boat, or the road for the bus). In this case, the model navigates from the surrounding context to the object itself. *(b)* The contextual region may include the target (*e.g.* the tree branches in which the bird is hiding), and inform the model to fixate a subregion likely to represent it. This situation corresponds to a coarse-to-fine search for the target. *(c)* Finally, the target may be too big, and fixations inside its region may initially not foveate it sufficiently for an observation to be made. In such cases, object sub-parts are often chosen as evidence regions to guide the search to other subparts (*e.g.* the various features of the front of the railway engine), until the object is included in the observation set and therefore the confidence function is evaluated on its entire extent. In this case, the model behavior resembles a perceptual grouping process in which smaller scale parts are integrated to deduce the extent of a large object.

5. Conclusions

We have presented a reinforcement learning model for visual object detection. In contrast to methods that operate exhaustively over a hypothesis space, we have derived a fully trainable sequential model that can efficiently sample only a few image locations in order to accumulate evidence on the target location. Our model is image and category specific and can explicitly balance the trade-off between exploration (improving accuracy) and exploitation (efficiently terminating search when sufficient evidence has been gathered). Our methodology is general and applicable to any detector response function. We report encouraging results in the PASCAL VOC 2012 object detection dataset, showing that the proposed methodology achieves almost *two orders of magnitude speed-up* over sliding window methods.

Acknowledgements. Work partly supported by CNCS-UEFISCDI under PCE-2011-3-0438, JRP-RO-FR-2014-16.

References

[1] D. Banica and C. Sminchisescu. Second-Order Constrained Parametric Proposals and Sequential Search-Based Structured Prediction for Semantic Segmentation in RGB-D Images. *CVPR*, 2015. 2

[2] L. Bazzani, d. N. Freitas, H. Larochelle, and V. Muriono. Learning attentional policies for tracking and recognition in video with deep networks. In *ICML*, 2011. 2

[3] N. J. Butko and J. R. Movellan. Infomax control of eye movements. *IEEE TAMD*, 2(2):91–107, 2010. 2

[4] J. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *ICCV*, 2015. 2, 6

[5] J. Carreira and C. Sminchisescu. CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts. *PAMI*, 2012. 2, 7

[6] M. Choi, J. Lim, A. Torralba, and A. Willsky. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010. 2

[7] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009. 2

[8] G. Dulac-Arnold, L. Denoyer, N. Thome, M. Cord, and P. Gallinari. Sequentially generated instance-dependent image representations for classification. *ICLR*, 2014. 2

[9] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014. 2

[10] M. Everingham, L. V. Gool, C. K. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/. 7

[11] P. F. Felzenszwalb, R. B. Girschick, and D. McAllester. Cascade object detection with deformable part models. In *CVPR*, 2010. 1

[12] R. Girschick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 7

[13] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari. An active search strategy for efficient object class detection. In *CVPR*, 2015. 2, 8

[14] B. Goodrich and I. Arel. Reinforcement learning based visual attention with application to face detection. In *CVPR*, 2012. 2, 6

[15] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *ICCV*, 2009. 1

[16] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008. 2

[17] L. Itti, G. Rees, and J. K. Tsotsos. *Neurobiology of attention*. Elsevier, 2005. 1

[18] S. Karayev, T. Baumgartner, M. Fritz, and T. Darrell. Timely object recognition. In *NIPS*, 2012. 2, 8

[19] S. Karayev, M. Fritz, and T. Darrell. Anytime recognition of objects and scenes. In *CVPR*, 2014. 2

[20] I. Kokkinos. Shufflets: Shared mid-level parts for fast object detection. In *ICCV*, 2013. 2

[21] P. Krahenbuhl and V. Koltun. Learning to propose objects. In *CVPR*, 2015. 2

[22] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 7

[23] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008. 1

[24] H. Larochelle and G. E. Hinton. Learning to combine foveal glimpses with a third-order boltzmann machine. In *NIPS*, 2009. 2

[25] G. F. Lucas Paletta and C. Seifert. Q-learning of sequential attention for visual object recognition from informative local descriptors. In *ICML*, 2005. 2

[26] S. Mathe, A. Pirinen, and C. Sminchisescu. Deep reinforcement learning methods for visual object detection. Technical report, Lund University, 2016. 2, 3

[27] S. Mathe and C. Sminchisescu. Action from still image dataset and inverse optimal control to learn task specific visual scanpaths. In *NIPS*, 2013. 2

[28] S. Mathe and C. Sminchisescu. Multiple instance reinforcement learning for efficient weakly-supervised detection in images. *CoRR*, abs/1412.0100, 2014. 2

[29] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In *NIPS*, 2014. 2

[30] M. Pedersoli, A. Vedaldi, and J. Gonzales. A coarse-to-fine approach for fast deformable object detection. In *CVPR*, 2011. 1

[31] A. Rabinovich, A. Vedaldi, C. Calleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007. 2

[32] P. Rantalankila, J. Kannala, and E. Rahtu. Generating object segmentation proposals using global and local search. In *CVPR*, 2014. 2

[33] R. Shaoqing, H. Kaiming, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*, 2015. 7

[34] N. Shapovalova, M. Raptis, L. Sigal, and G. Mori. Action is in the eye of the beholder: Eye-gaze driven model for spatio-temporal action localization. In *NIPS*, 2013. 1

[35] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005. 2

[36] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998. 2, 6

[37] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *NIPS*, 2013. 2

[38] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 104, 2013. 2, 8

[39] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009. 1

[40] Y. Wei and L. Tao. Efficient histogram-based sliding window. In *CVPR*, 2010. 2

[41] R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992. 6